**Use word embeddings to classify Twitter sentiment using Azure Machine Learning and Team Data Science Process**

We have recently published a sample to show how to use Azure Machine Learning together with Team Data Science Process (TDSP) to execute an AI project for Twitter sentiment classification. This blog provides a summary of that published sample, which is publicly available through a GitHub repository.

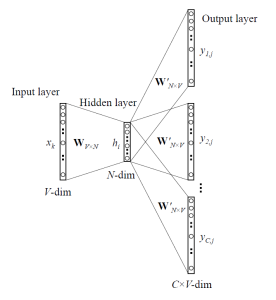
We demonstrated the usage of Word Embedding algorithms like Word2Vec and Sentiment Specific Word Embedding (SSWE) to predict Twitter sentiment. We following Team Data Science Process to execute this project. The data used in this project is Sentiment140 dataset, which contains the actual content of the tweet (with emoticons removed) along with the polarity of each of the tweet (positive and negative, neutral tweets are removed for this project). Sentiment140 dataset has been labelled using the concept of distant supervision as explained in the paper [Twitter Sentiment Classification Using Distant Supervision](http://cs.stanford.edu/people/alecmgo/papers/TwitterDistantSupervision09.pdf). This project is executed using TDSP templates which consist of the following parts:

* Data acquisition and understanding
* Modeling
  + Feature Engineering
  + Model Creation
  + Model Evaluation
* Deployment

Some highlights from this sample:

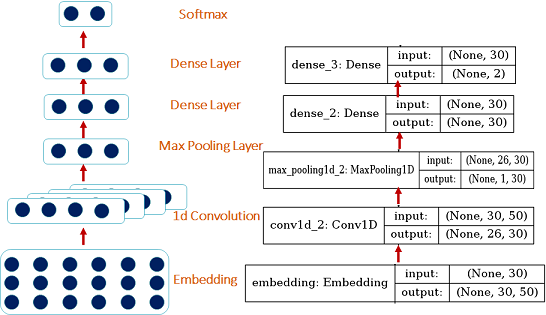
* This sample is running on the latest [Azure Machine Learning Workbench](https://docs.microsoft.com/en-us/azure/machine-learning/preview/), which is currently in public preview.
* Model training is performed in Azure Data Science Virtual Machine with GPU.
* Word embeddings using Word2Vec and SSWE are generated for modeling.
* Deep learning frameworks and packages such as TensorFlow, CNTK and Keras are applied in this project.
* Four models using different word embedding methods and classification modeling techniques are trained and compared.
* The most accurate trained model is deployed to a web service using [Azure Container Service](https://azure.microsoft.com/en-us/services/container-service/).

The Word2Vec algorithm is based on the the paper [Mikolov, Tomas, et al. Distributed representations of words and phrases and their compositionality. Advances in neural information processing systems. 2013.](https://arxiv.org/abs/1310.4546) Skip-gram is a shallow neural network taking the target word encoded as a one hot vector as input and using it to predict nearby words. The skip-gram based architecture is shown in the following figure.

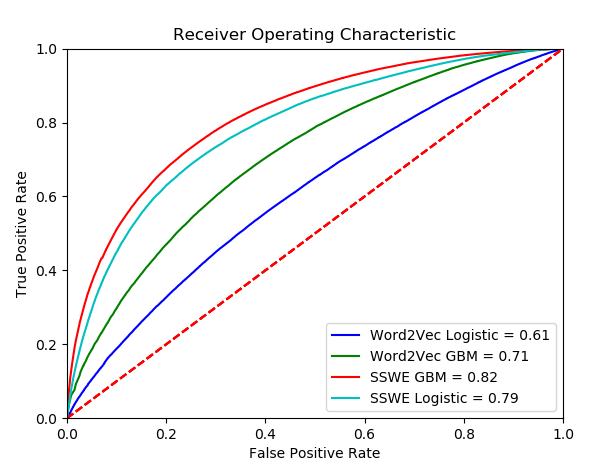


Sentiment Specific Word Embedding (SSWE) Algorithm proposed in [Tang, Duyu, et al. "Learning Sentiment-Specific Word Embedding for Twitter Sentiment Classification." ACL (1). 2014.](http://www.aclweb.org/anthology/P14-1146) tries to overcome the weakness of Word2vec algorithm that the words with similar contexts and opposite polarity can have similar word vectors. This means that Word2vec may not perform very accurately for the tasks like sentiment analysis. SSWE algorithm tries to handle this weakness by incorporating both the sentence polarity and the word's context in to its loss function.

We are using a variant of SSWE in this sample. SSWE uses both the original ngram and corrupted ngram as input and it uses a ranking style hinge loss function for both the syntactic loss and the semantic loss. Ultimate loss function is the weighted combination of both the syntactic loss and semantic loss. For the purpose of simplicity, we are using only the semantic cross entropy as the loss function. As we are going to see later on, even with this simpler loss function the performance of the SSWE embedding is better than the Word2Vec embedding as evaluated by the accuracy of the sentiment classification. SSWE inspired neural network model that we use in this sample is shown in the following figure



We compared Logistic Regression and Gradient Boosted Decision Tree models using Word2Vec and SSWE as features, results show that Gradient Boosted Tree Model with SSWE embedding performs the best. Then this is model is deployed to a web service using Azure Container Service.



For details, please refer to this [article](https://docs.microsoft.com/azure/machine-learning/preview/scenario-tdsp-twitter-sentiment). We also provided all the scripts and a detailed [walkthrough](https://github.com/Azure/MachineLearningSamples-TwitterSentimentPrediction/blob/master/docs/deliverable_docs/Step_By_Step_Tutorial.md) in this [GitHub repository](https://github.com/Azure/MachineLearningSamples-TwitterSentimentPrediction).

We would love to hear your feedback on this sample – you can send us your feedback and comments via the GitHub [issues page](https://github.com/Azure/MachineLearningSamples-TwitterSentimentPrediction/issues).